Aᵀ 2154 (handwritten)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:

        Doron Juster et al.

Serial No.:	09/533,468	Group Art Unit:	2154
Filed:	March 23, 2000	Examiner:	Sindya Narayanaswamy
For:	Local Queue Creation Security		

APPELLANT'S BRIEF ON APPEAL

Mail Stop Appeal Brief—Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

Dear Sir:

    This Appeal Brief is submitted in support of Applicants' appeal from the Final Office action of February 25, 2004. The corresponding Notice of Appeal was filed on June 1, 2004. This is a revised version of Applicants' Appeal Brief originally filed on November 1, 2004 which is being submitted in response to a Notification of Non-Compliant Appeal Brief mailed February 11, 2005. This version of Applicants' Appeal Brief addresses the deficiency raised in the Notification.

In re Application of: Juster et al.
Application No.: 09/533,468

I. The Real Party in Interest

Microsoft Corporation of Redmond, Washington, is the real party in interest for this appeal. The application was assigned to Microsoft by the inventors on March 13, 2000, and that assignment was recorded at the Patent and Trademark Office ("PTO") at Reel 010687, Frame 0468.

II. Related Appeals and Interferences

No appeals or interferences related to this appeal are known to the appellant, to the appellant's legal representatives, or to the appellant's assignee.

III. Status of the Claims

This application was filed with 16 claims. The Final Office Action dated February 25, 2004, rejected all of the pending claims, i.e., claims 1 through 16, and these claims are at issue in this appeal.
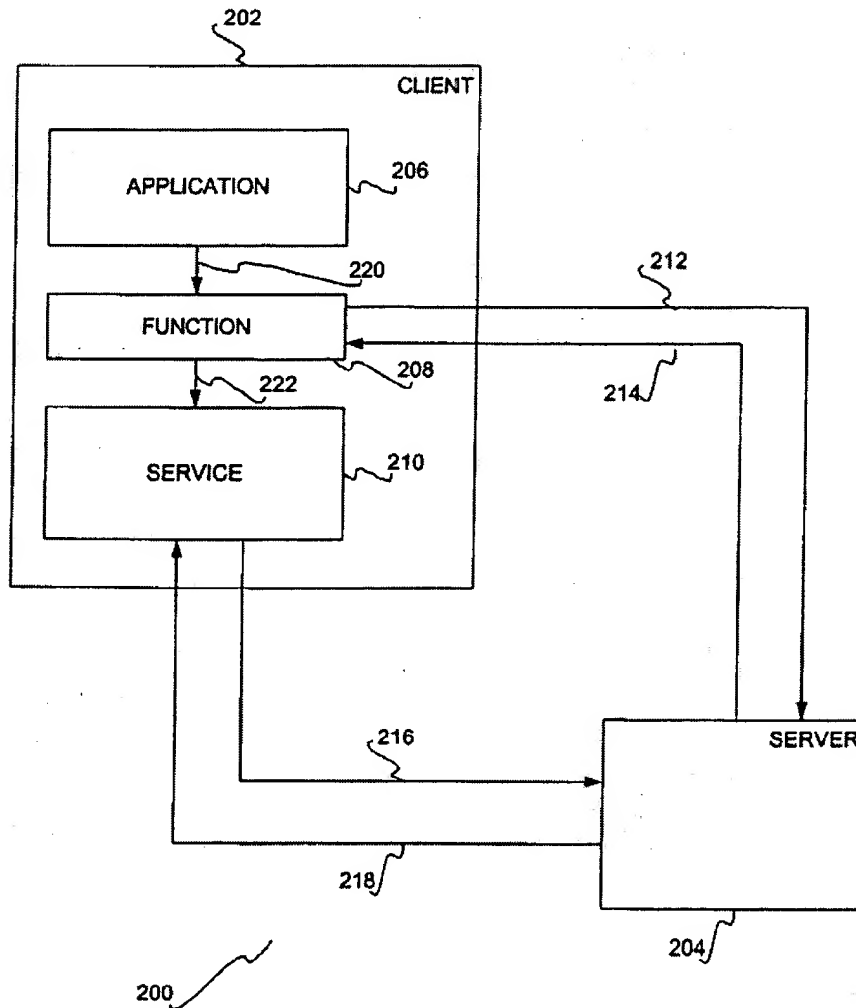
IV. Status of Amendments

No amendment was filed in response to the Final Office action and, thus, there is no outstanding amendment in this application.

V. Summary of the Invention

The present application is directed to a method for enforcing security on the creation of queues at a local machine or local computing environment, which improves on the user-based security of known systems. In one embodiment, a method includes sending a first request to create a local queue, by an application of a client from a function of the client to a server. If the server determines that the user under which the application is running has permission to create local queues, the local queue is created. Otherwise, a second request to create the local queue is sent from the function of the client to a server having permission to create local queues. If the service

2

determines that the second request originated locally – i.e., within the client itself – then the service

calls the server to create the local queue.

FIG 2

Embodiments of the invention provide for local-based security within message transaction

systems, even where the underlying operating system has only user-based security. In the method

described in the previous paragraph, for example, the user may now have by default no permission

to create local queues. Thus, when the function calls the server, the server will deny the request to

create a local queue. However, the service running on the client does have user-level security to create queues, but it only allows queues to be created for requests to create queues that originate from within the client. Thus, when the function calls the service, if the service determines that this request to create a queue originated locally, only then does it call the server to create a queue. From the server's perspective, it is still enforcing user-level security, where both the user himself or herself and the service have such security. However, the service only permits queues to be created that are on the same client from which the request originated – thus, the service itself provides for local-machine or local-based security by its enforcement as to what situations in which it allows queues to be created. The invention includes computer-implemented methods, machine-readable media, computerized systems, and computers of varying scopes.

Turning to the claims, and with specific reference to the independent claims (claims 1, 8, and 12), claim 1 recites a method comprising: *sending a first request to create a local queue by an application of a client from a function of the client to a server*;

> Referring to FIG. 3, a flowchart of a method according to an embodiment of the invention is shown. In 300, a first request is sent to create a local queue from a function of a client to a server. The request originated from an application of the client, which sent the request to the function. The client and the server are desirably part of a message transaction system implemented in conjunction with an operating system that provides only for user-level security.

See Application, page 12, lines 14-19.

upon determining at the server that a user under which the application is running has permission to create local queues, creating the local queue by the server;

> In 302, the server determines whether the user under which the application is running has permission to create local queues. In one embodiment, the default user rights are such that no such permission is given - that is, users cannot by default create local queues. However, if the user does have permission to create a local queue, then the server creates the queue in 304, and the method is finished. However, more usually, the method instead proceeds to 306, since the user typically does not have permission to create local queues.

See Application, page 12, lines 20-23; page 13, lines 1-3.

otherwise, sending a second request to create the local queue by the application of the client

from the function of the client to a service having permission to create local queues;

> In 306, the function has received indication that the request to create a queue was denied, and therefore sends a second request to create the local queue to a service running on the client. The service in one embodiment desirably has user-level rights to create local queues. In 308, the service determines whether the request received originated locally. That is, the service determines whether the request is from within the client on which it is running itself, and whether the request pertains to the creation of a queue on the client. If not, then the request is denied in 310, and the method is finished.

See Application, page 13, lines 4-10.

and only upon determining by the service that the second request originated locally, calling

the server by the service to create the local queue.

> Otherwise, the method proceeds to 312, where the service calls the server to create the queue. It is noted that the method then proceeds back to 302, where the server determines whether the user has user-level permission to create the queue. However, since desirably the service is its own "user," and has user-level permission to create local queues, the server grants the request, and creates the queue in 314, where the method is finished.

See Application, page 13, lines 11-16.

Claim 1:   A computer-implemented method comprising:

sending a first request to create a local queue by an

application of a client from a function of the client to a server;

upon determining at the server that a user under which the

application is running has permission to create local queues, creating

the local queue by the server;

otherwise, sending a second request to create the local queue

by the application of the client from the function of the client to a

service having permission to create local queues; and

only upon determining by the service that the second request

originated locally, calling the server by the service to create the local

queue.

Claim 8 is a computer-readable medium or "Beauregard" counterpart to claim 1 and

contains language substantially similar to that of claim 1:

> In some embodiments, the methods are computer-implemented. The computer-implemented methods
> can be realized at least in part as one or more programs running on a computer – that is, as a program
> executed from a computer-readable medium such as a memory by a processor of a computer, such as
> the computer shown in and described in conjunction with FIG. 1. The programs are desirably storable
> on a machine-readable medium such as a floppy disk or a CD-ROM, for distribution and installation
> and execution on another computer.

See Application, page 12, lines 3-9.

Claim 8:   A machine-readable medium having instructions stored thereon for

execution by a client processor to perform a method comprising:

sending a first request to create a local queue by an

application of the client from a function of the client to a server;

upon receiving indication from the server that the first request

was denied, sending a second request to create the local queue by the

application of the client from the function of the client to a service

having permission to create local queues; and,

only upon determining by the service that the second request

originated locally, calling the server by the service to create the local

queue.

Claim 12 is an apparatus (system) claim which also contains language substantially similar

to that of claim 1:

In this section of the detailed description, systems according to varying embodiments of the invention are described. The systems are described in conjunction with FIG, 2, which is a diagram of a system according to an embodiment of the invention.

The methods can be implemented in conjunction with the system of FIG. 2 described in the previous section of the detailed description, such as in conjunction with the application program, the function, the service, and the server of FIG, 2, although those of ordinary skill within the art can appreciate that the invention itself is not so limited.

See Application, page 9, lines 2-5; page 12, lines 9-13.

Claim 12:   A computerized system comprising:

a server having the capability to create local queues; and,

a client comprising:

an application program;

a function designed to send a first request to create a local queue by the application program to the server and to receive indication from the server as to whether the first request was denied; and,

a service having permission to create local queues and designed to receive a second request to create the local queue by the application from the function, the function sending the second request upon receiving indication from the server that the first request was denied, the service calling the server to create the local queue only upon determining that the second request originated locally.

VI.   Issues on Appeal

The issue on appeal is whether claims 1 through 16 are unpatentable under 35 U.S.C. § 103(a) as obvious over the combination of Applicant's Admitted Prior Art and U.S. Patent No.

In re Application of: Juster et al.
Application No.: 09/533,468

6,269,399 ("Dyson"). Specifically, the issue is whether this combination teaches all of the elements of these claims.

VII. Grouping of the Claims

The arguments presented in the appeal brief apply equally to all of the presently rejected claims. Accordingly, pending claims 1 through 16 stand or fall together.

VIII. Argument

To present a prima facie case of obviousness under 35 U.S.C. § 103(a), the cited references must, either separately or in combination, suggest or teach all of the elements of the rejected claims. *See* the Manual of Patent Examining Procedure § 2143.

Applicants respectfully submit that the Final Office action has failed to establish a prima facie case of obviousness because the combination of cited references neither teaches nor suggests all of the elements of the rejected claims. Therefore, reconsideration and allowance of claims 1 through 16 are respectfully solicited.

A. The Rejections

The Final Office action dated February 25, 2004, rejected all pending claims under 35 U.S.C. § 103(a) as obvious over a combination of Applicant's Admitted Prior Art ("AAPA") and Dyson.

The Final Office action asserts that AAPA teaches a method for sending a first request to create a local queue by an application of a client from a function of the client to a server and creating the local queue when it is determined by the server that the user has permission to create local queues. However, AAPA contemplates only a client-centric system for creating local queues and, as the Final Office action pointed out, AAPA does not "specifically teach the method of

sending second requests to a service having permission to create local queues and the method of the service determined that the 2nd request originated locally, calling the server by the service to create local queues." *See* the Final Office action, page 2, paragraph 5. In terms of the pending independent claims 1, 8, and 12, the Final Office action admitted that AAPA does not teach the emphasized elements:

Claim 1:   A computer-implemented method comprising:

         sending a first request to create a local queue by an application of a client from a function of the client to a server;

         upon determining at the server that a user under which the application is running has permission to create local queues, creating the local queue by the server;

         otherwise, sending a second request to create the local queue by the application of the client from the function of the client to a service having permission to create local queues; and

         only upon determining by the service that the second request originated locally, calling the server by the service to create the local queue.

(Emphasis added.) (Claim 8 is a computer-readable medium or "Beauregard" counterpart to claim 1 and contains language substantially similar to that of claim 1. Claim 12 is an apparatus claim which also contains language substantially similar to that of claim 1. Claims 8 and 12 were summarily rejected in the Final Office action "under the same reasoning as claim 1..." *See* the Final Office action, page 4, paragraph 14.)

To supply the admitted lack of disclosure of these recited claim elements by AAPA, the Final Office action turns to Dyson. Dyson teaches a system and associated method for securely exchanging information between first and second parties, the first party being a wholesale of telecommunications services, such as wireless communications services, and the second party being a reseller of the telecommunications services of the first party. The system of Dyson includes at least a first queue system associated with the first party for securely transferring messages to a second queue system accessible by the second party on corresponding queues, a firewall for securely routing the messages, and a processor for validating the messages in accordance with a contract between the first and second parties, the contract having provisions directed to the provision of such telecommunications services. Based on Dyson's disclosure of a first queue and a second queue, the Final Office action considered that Dyson teaches a "method of sending second requests to a service having permission to create local queues and when the service determined that the 2nd request originated locally, calling the server by the service to create local queues." The Final Office action then presented the combination of AAPA and Dyson as teaching all the elements of the pending claims.

B.   The Final Office Action Failed to Establish a Prima Facie Case of Obviousness

The combination of AAPA and Dyson does not teach all the elements of the independent claims, that is, of claims 1, 8, and 12: "1.) sending a first request to create a local queue by an application of a client from a function of the client to a server; 2.) upon determining at the server that a user under which the application is running has permission to create local queues, creating the local queue by the server; 3.) otherwise, sending a second request to create the local queue by the application of the client from the function of the client to a service having permission to create local queues; and 4.) only upon determining by the service that the second request originated locally, calling the server by the service to create the local queue."

The Final Office action cites page 1, line 17 through page 2, line 2 in the application as the

AAPA disclosing, "sending a first request to create a local queue by an application of a client from a

function of the client to a server and creating the local queue when determined by the server that the

user has permissions." *See* the Final Office action, page 2, paragraph 5. The excerpt cited by the

Final Office action is reproduced below:

> Frequently, the operating systems in conjunction with which message transaction systems are
> implemented have security that is user-based. This means that a given user, if he or she has
> permission to create queues, for example, is able to create queues regardless of the client onto which
> the user is actually logged. Generally, in such systems, users have default permission to create
> queues, since the ability for users to create queues on at least their local machines is necessary for
> applications within the message transaction systems to run properly.

It would appear that the above passage merely discloses a system where a user that has permission

to create queues is able to create queues on any client, as the state of the art is to grant users

permission to create queues by default. This reading is consistent with the portion of the application

immediately subsequent to the one cited by the Final Office action:

> However, this situation can lead to compromised security. Because *the security is user-based, and by
> default the user is able to create queues on any client*, a malicious user can swamp a given client by
> requesting too many to be created queues on the client – thus denying service for legitimate users.
> The security is thus not local- or local machine-based – *because of the underlying operating system on
> which the message transaction system is running, within the prior art, a given user cannot be limited
> to creating queues only on the specific client the user is logged onto.* Once the user is given
> permission to create queues, due to the user-based security, the user is able to create queues on any
> client. For this and other reasons, there is a need for the present invention.

(Emphasis added.) *See* Application, page 2, lines 3-11. Thus the deficiencies of the prior art

detailed in the application are clear – users are given permission to create queues by default and thus

can create queues not only on the client machine that they are logged onto, but also any other client

as well.

Equally clear is that the present invention addresses the shortcomings of the prior art in a

number of ways. The first is by *"sending a first request to create a local queue,* by an application of

a client from a function of the client *to a server.* If *the server determines that the user under which

the application is running has permission to create local queues,* the local queue is created."

In re Application of: Juster et al.
Application No.: 09/533,468

(Emphasis added.) *See* Application, page 2, lines 14-17. Hence, the limitations in claim 1 (and analogous limitations in claims 8 and 12) of *"sending a first request to create a local queue* by an application of a client from a function of the client *to a server; upon determining at the server that a user under which the application is running has permission to create local queues,* creating the local queue by the server." Nowhere in AAPA is such a scheme to verify that the user has permission to create local queues by sending a request to the server even remotely disclosed.

While AAPA discloses a system where a user that has permission to create queues is able to create queues on any client, neither AAPA nor Dyson teach sending a second request to create the local queue by the application of the client from the function of the client to a service having permission to create local queues; and only upon determining by the service that the second request originated locally, calling the server by the service to create the local queue. The Final Office action acknowledges the lack of any such teaching in AAPA on page 2 at paragraph 5. However, the Final Office action does cite Dyson, specifically Figures 1 and 3A, for these elements, but there Dyson merely discloses a first queue and a second queue rather than the claimed "sending a second request to create the local queue" and "upon determining by the service that the second request originated locally, calling the server by the service to create the local queue."

Indeed, when Applicants' representatives took the opportunity after the first Office action to conduct a telephonic interview with Examiners Narayanaswamy and Maung on June 19, 2003 – in an effort to clarify that the Dyson patent does not disclose or suggest creating queues – Examiner Maung admitted that the Dyson patent does not teach creating queues. Examiner Maung countered that Dyson uses queues and, therefore, the queues are necessarily created prior to their use. For the sake of argument, Applicants agreed that the queues are of course created at some point in time

12

In re Application of: Juster et al.
Application No.: 09/533,468

prior to when the teachings of Dyson are implemented, but their creation under the circumstances
required by the claims is neither described nor suggested by the Dyson patent.[1]

It is less than clear as to how the Dyson reference could possibly teach "sending a second
request to create the local queue by the application of the client from the function of the client to a
service having permission to create local queues; and only upon determining by the service that the
second request originated locally, calling the server by the service to create the local queue" as
required by the claims, when Dyson *does not even explicitly disclose or teach creation of queues*.
The Final Office action cites to column 6, lines 20-31 in the Dyson patent to assert that Dyson
teaches creating a queue on a conditional basis. *See* Final Office action, page 6, paragraph 3. The
excerpt cited by the Final Office action is reproduced below:

> In such circumstances, the step of validating includes determining which of the particular resellers
> (e.g., second party) on contract with the first party are affected by the particular outage. In this
> regard, at least a first message may be then be generated and sent (e.g., transmitted) to the affected
> resellers, such as the second party, via, for example, the above-identified pair of queues of the first
> and second queue systems associated with the first and second parties. In this regard, the present
> invention facilitates the exchange of information affecting service to subscribers of the second
> party.

Clearly absent from the above passage is any teaching or suggestion of sending a second request to
create the local queue by the application of the client from the function of the client to a service
having permission to create local queues; and only upon determining by the service that the second
request originated locally, calling the server by the service to create the local queue. What the cited
section merely states is that *a message may be generated and sent via the first and second queues*.
Nowhere in this passage, or anywhere else in Dyson for that matter, are local queues created in
response to sending a second request to create the local queue by the application of the client from

---

[1] Applicants also disagree with Examiner Narayanaswamy's characterizations of Applicants' statements in their
Response to Office Action Mailed March 28, 2003. The Final Office action, on page 6 at paragraph 3 states that,
"Applicant has admitted (on both pages 3 and 4) that, the Dyson patent may suggest creating the queues in the patent
since the queues must be created before they can be used." Applicants' use of qualifying terms such as "may" and
"At most" on pages 3 and 4 of their response indicate that the positions are being taken *in arguendo* and should not
interpreted to mean that Applicants agree with Examiner's interpretation of the Dyson reference.
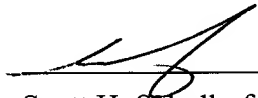
In re Application of: Juster et al.
Application No.: 09/533,468

the function of the client to a service having permission to create local queues; and only upon

determining by the service that the second request originated locally.

Because it did not show a combination that teaches all the elements of claims 1, 8, and 12,

the Final Office action failed to present a prima facie rejection of claims 1, 8, and 12. Therefore,

these claims should be allowed. The remaining pending claims, that is, claims 2 through 7, 9

through 11, and 13 through 16, all depend upon claims 1, 8, and 12 respectively and are thus

allowable for at least the same reasons that claims 1, 8 and 12 are allowable.

In view of the above, Applicants submit that the Final Office action failed to establish that

claims 1 through 16 are obvious in light of the cited art. Accordingly, these claims should be

allowable, and applicant respectfully solicits the Board to consider this Appeal, to remove the

outstanding grounds of rejection, and to allow claims 1 through 16.

Respectfully submitted,

Scott H. Schulhof, Reg. No. 53,568
One of the Attorneys for Applicant
LEYDIG, VOIT & MAYER, LTD.
Two Prudential Plaza, Suite 4900
180 North Stetson
Chicago, Illinois 60601-6780
(312)616-5600 (telephone)
(312)616-5700 (facsimile)

Date: March 11, 2005

Appendix: The Claims on Appeal

1. A computer-implemented method comprising:

   sending a first request to create a local queue by an application of a client from a function of the client to a server;

   upon determining at the server that a user under which the application is running has permission to create local queues, creating the local queue by the server;

   otherwise, sending a second request to create the local queue by the application of the client from the function of the client to a service having permission to create local queues; and

   only upon determining by the service that the second request originated locally, calling the server by the service to create the local queue.

2. The method of claim 1, wherein the server uses only user-level security.

3. The method of claim 1, wherein the service, by calling the server to create the local queue only upon determining that the second request originated locally, provides for local-level security.

4. The method of claim 1, wherein the service comprises a transactional message service.

5. The method of claim 1, wherein the function is accessed by the application via an application programming interface (API) of the function.

6. The method of claim 1, wherein the service is running on the client.

15

7. The method of claim 1, wherein the user by default lacks permission to create local queues.

8. A machine-readable medium having instructions stored thereon for execution by a client processor to perform a method comprising:

sending a first request to create a local queue by an application of the client from a function of the client to a server;

upon receiving indication from the server that the first request was denied, sending a second request to create the local queue by the application of the client from the function of the client to a service having permission to create local queues; and,

only upon determining by the service that the second request originated locally, calling the server by the service to create the local queue.

9. The medium of claim 8, wherein the server uses only user-level security, and the service, by calling the server to create the local queue only upon determining that the second request originated locally, provides for local-level security.

10. The medium of claim 8, wherein the service comprises a transactional message service, the function is accessed by the application via an application programming interface (API) of the function, and the service is running on the client.

11. The medium of claim 8, wherein the user by default lacks permission to create local queues.

12.   A computerized system comprising:

        a server having the capability to create local queues; and,

        a client comprising:

                an application program;

                a function designed to send a first request to create a local queue by the

        application program to the server and to receive indication from the server as to

        whether the first request was denied; and

                a service having permission to create local queues and designed to receive

        a second request to create the local queue by the application from the function, the

        function sending the second request upon receiving indication from the server that

        the first request was denied, the service calling the server to create the local queue

        only upon determining that the second request originated locally.

13.   The system of claim 12, wherein the client further comprises a computer-readable

        medium and a processor, such that at least one of the application program, the function,

        and the service is executed by the processor from the medium.

14.   The system of claim 12, wherein the server uses only user-level security, and the service,

        by calling the server to create the local queue only upon determining that the second

        request originated locally, provides for local-level security.

15.   The system of claim 12, wherein the service comprises a transactional message service,

        and the function is accessed by the application via an application programming interface

        (API) of the function.

16. The system of claim 12, wherein the user by default lacks permission to create local queues.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:

Doron Juster et al.

Art Unit: 2154

Application No. 09/533,468

Examiner: Sindya Narayanaswamy

Filed: March 23, 2000

For: LOCAL QUEUE CREATION SECURITY

TRANSMITTAL OF
APPELLANTS' APPEAL BRIEF

Mail Stop Appeal Brief – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In accordance with 37 CFR 41.37, appellants hereby submit Appellants' Brief on Appeal.

The items checked below are appropriate:

1.   **Status of Appellants**

This application is on behalf of ☒ other than a small entity or ☐ a small entity.

2.   **Fee for Filing Brief on Appeal**

Pursuant to 37 CFR 41.20(2), the fee for filing the Brief on Appeal is for: ☒ other than a small entity or ☐ a small entity.

Brief Fee Due $500.00

3.   **Oral Hearing**

☐   Appellants request an oral hearing in accordance with 37 CFR 41.47.

A separate paper requesting oral hearing is attached.

4.   **Extension of Time**

☐   Appellants petition for a one-month extension of time under 37 CFR 1.136, the fee for which is $ 0.00.

☒   Appellants believe that no extension of time is required. However, this conditional petition is being made to provide for the possibility that appellants have inadvertently overlooked the need for a petition and fee for extension of time.

Extension fee due with this request: $

In re Appln. of Juster et al. Doron Juster et al.
Application No. 09/533,468 09/533,468

5. Total Fee Due

The total fee due is:

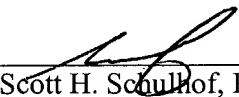|	
Brief on Appeal Fee	$500.00
Request for Oral Hearing	$ 0.00
Extension Fee (if any)	$ 0.00

Total Fee Due: $500.00

6. Fee Payment

☐ Attached is a check in the sum of $.

☒ Charge Account No. 12-1216 the sum of $500.00. A duplicate of this transmittal is attached.

7. Fee Deficiency.

☒ If any additional fee is required in connection with this communication, charge Account No. 12-1216. A duplicate copy of this transmittal is attached.

Respectfully submitted,

Scott H. Schulhof, Reg. No. 53,568
LEYDIG, VOIT & MAYER, LTD.
Two Prudential Plaza
180 North Stetson Ave., Suite 4900
Chicago, Illinois 60601-6780
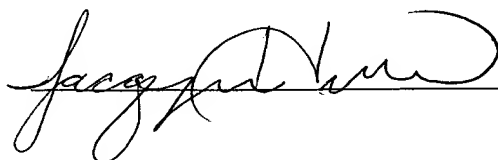(312) 616-5600 (telephone)
(312) 616-5700 (facsimile)

Date: March 11, 2005

CERTIFICATE OF MAILING

I hereby certify that this APPEAL BRIEF TRANSMITTAL AND APPEAL BRIEF (along with any documents referred to as attached or enclosed) is being deposited with the United States Postal Service on the date shown below with sufficient postage as first class mail in an envelope addressed to: Mail Stop Appeal Brief – Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450

Date: 3/11/05

Appeal Brief Transmittal (Revised 12/03/04)

2